JVM WORKSHOP – MODULE 1

FOR JFROG FREE-TIER

Who is speaking?

- **Sven Ruppert**
- DevSecOps, Java & Kotlin
- JFrog Inc
- **Twitter**: @SvenRuppert
- **eMail:** svenr@jfrog.com

Youtube: [DE] - bit.ly/Youtube-Sven
Youtube: [EN] - bit.ly/Outdoor-Nerd

THE AGENDA

BirdEyeView – Cloud Native

Difference between DevOps & DevSecOps

Why DevSecOps will minimize your risk

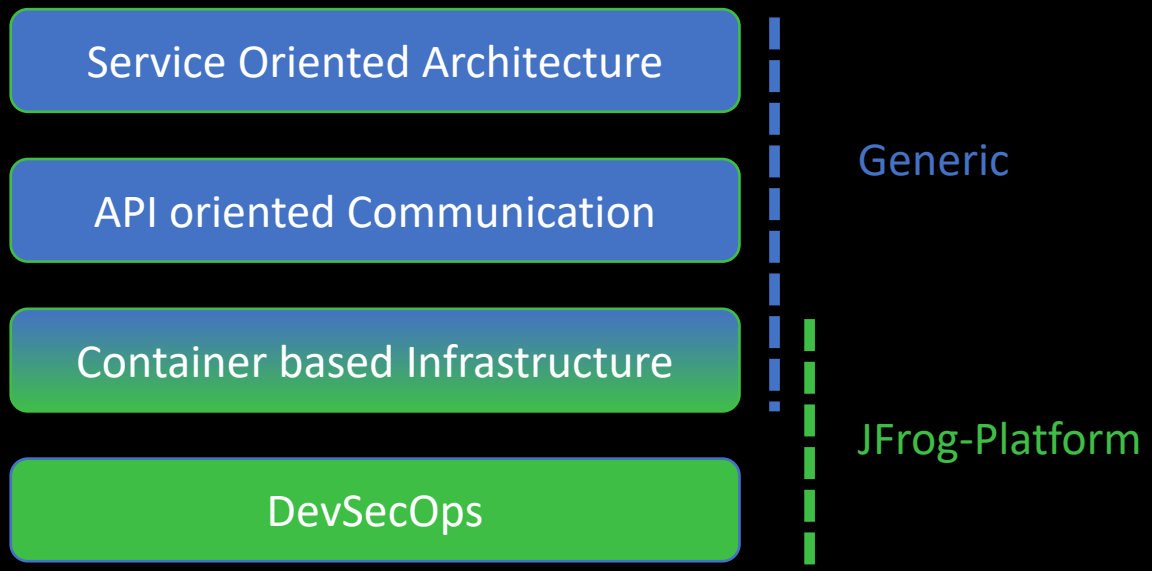
Security from scratch

SpeedUp your CI and true Immutability

The power of integration

BirdEye View – Cloud Native

BirdEye View – Cloud Native



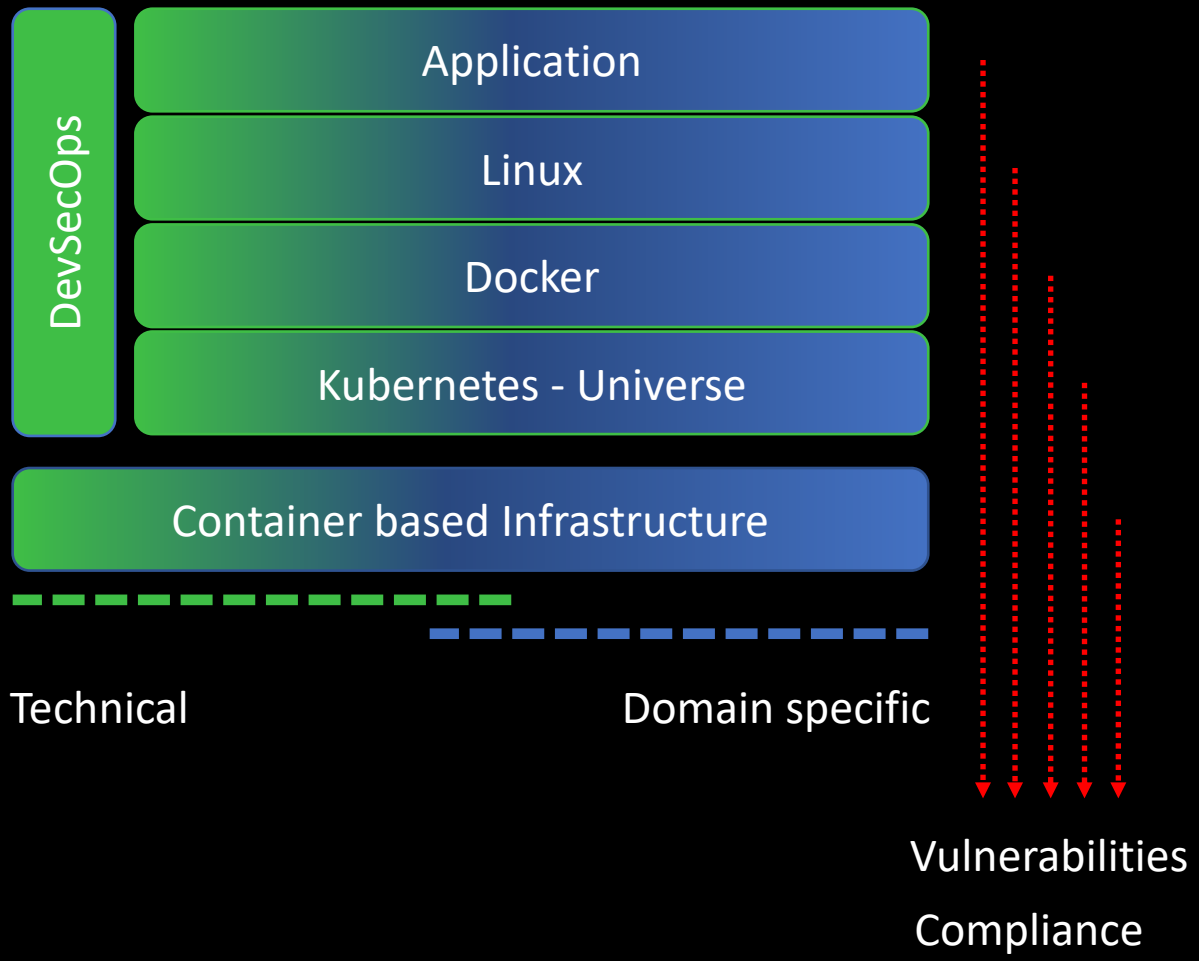
Shorter release cycles

Split into small and individual parts

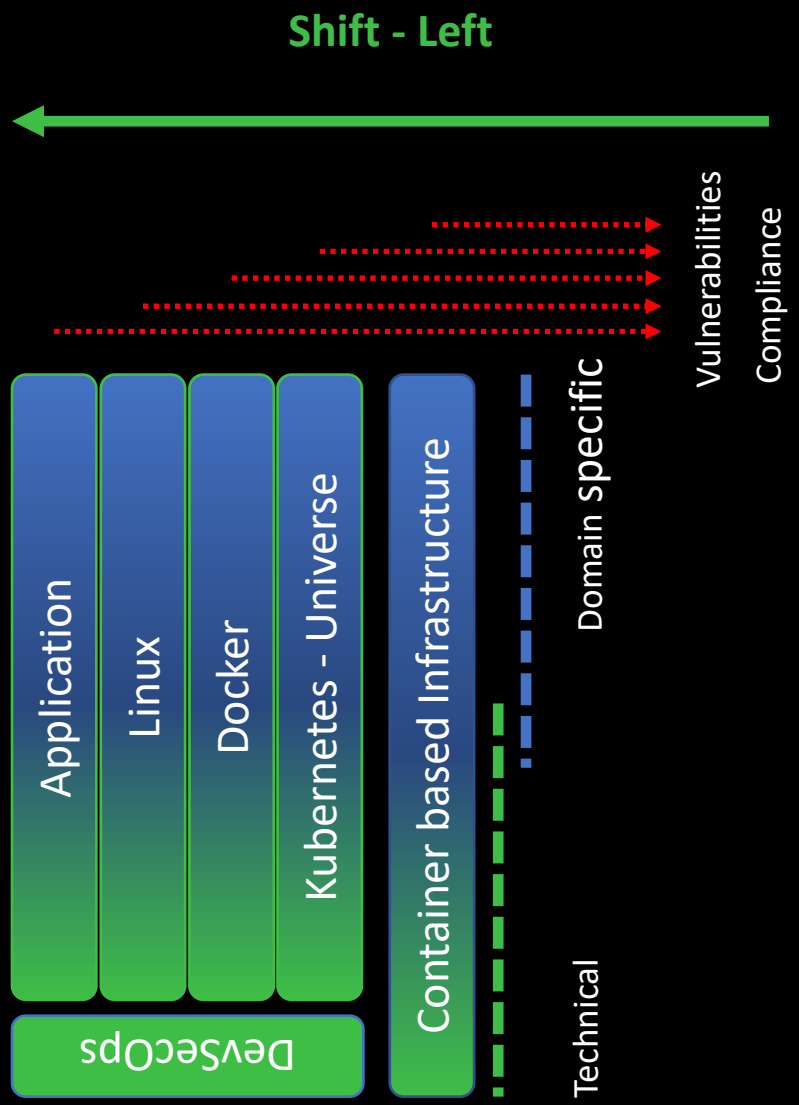
Shorter lifetime, earlier rewrite

Support for polyglot environment's

BirdEye View – Cloud Native



BirdEye View – Cloud Native



Difference between DevOps & DevSecOps

Difference between DevOps & DevSecOps

Pure DevOps

- Coding
- Building
- Testing
- Packaging
- Releasing
- Configuring
- Monitoring

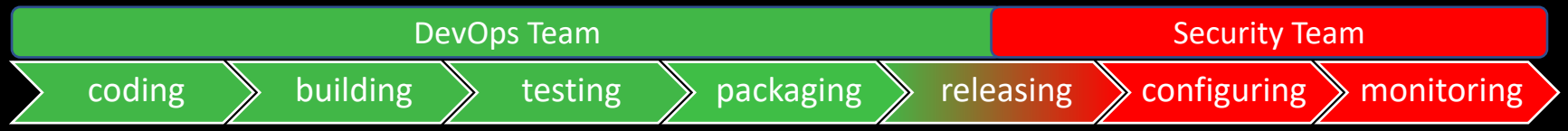
Difference between DevOps & DevSecOps

Developer Team versus Operations Team
Issues mostly found in production
Slow release process

Security Issues – high financial impact

Bugs – high financial impact

Difference between DevOps & DevSecOps



DevOps Team versus Security Team

Security is Black-Magic

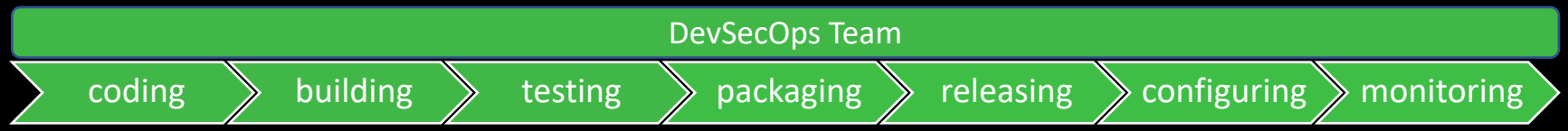
Security-Tests like second Release-Process

Security is a Process after the Sprint

Security Issues – high financial impact

Bugs – lower financial impact

Difference between DevOps & DevSecOps



Security Issues – low financial impact → WHY?

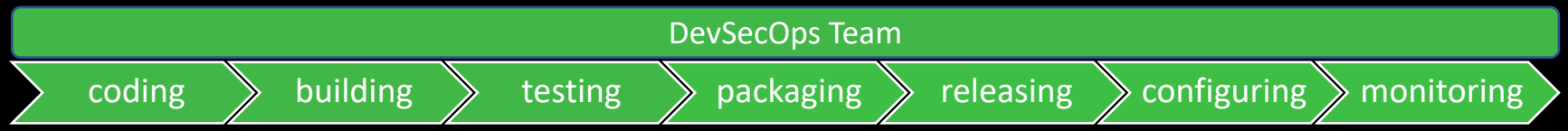
Bugs – low financial impact

What is the right place for SEC?

Is SEC just a product?

Will SEC slow me down?

Difference between DevOps & DevSecOps



#SecurityFirst - culture

Integrate Security practices inside the DevOps process

Introducing security as early as possible

Why DevSecOps will minimize your risk

Why DevSecOps will minimize your risk

Between 40% and 60 % of a project is based on Open Source

Focus on your UseCase

Don´t reenvent the wheel!

Easy to analyze compared to closed source

Security issues are found fast (mostly)

Many different OpenSource Licences

Don´t use the wrong one in your project

Check all transitive dependencies!

Why DevSecOps will minimize your risk

Automate as much as possible

Reduce the boring workload

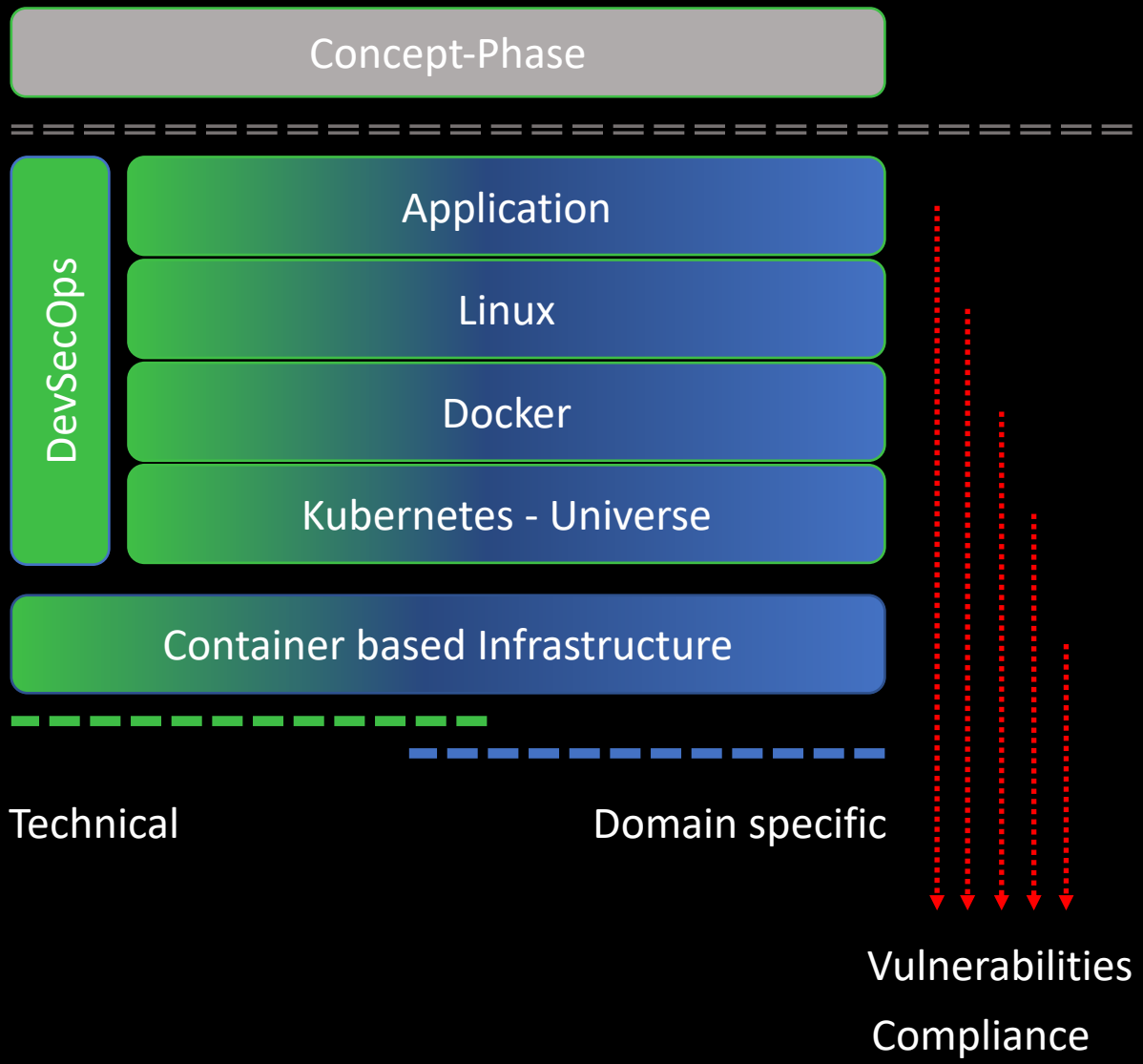
Be reproducible - everywhere

Kill as early all

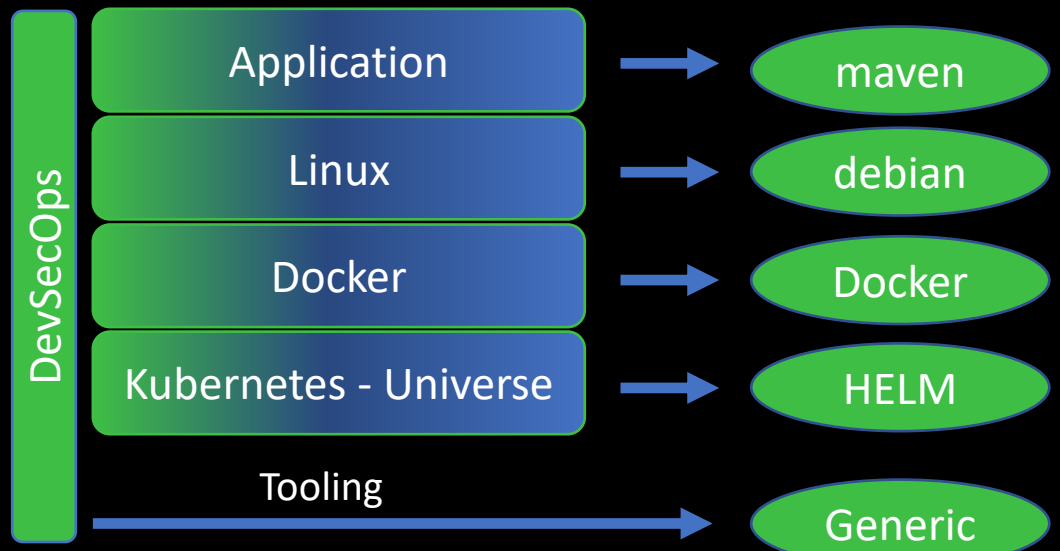
- bugs
- compliance-issues
- security-issues

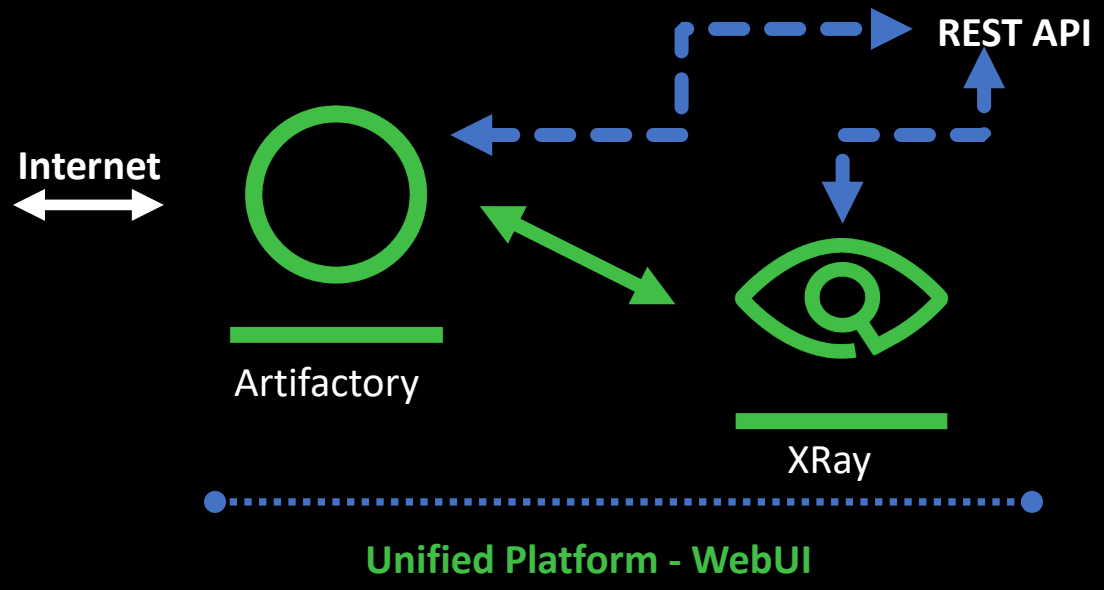
Security from scratch

Security from scratch

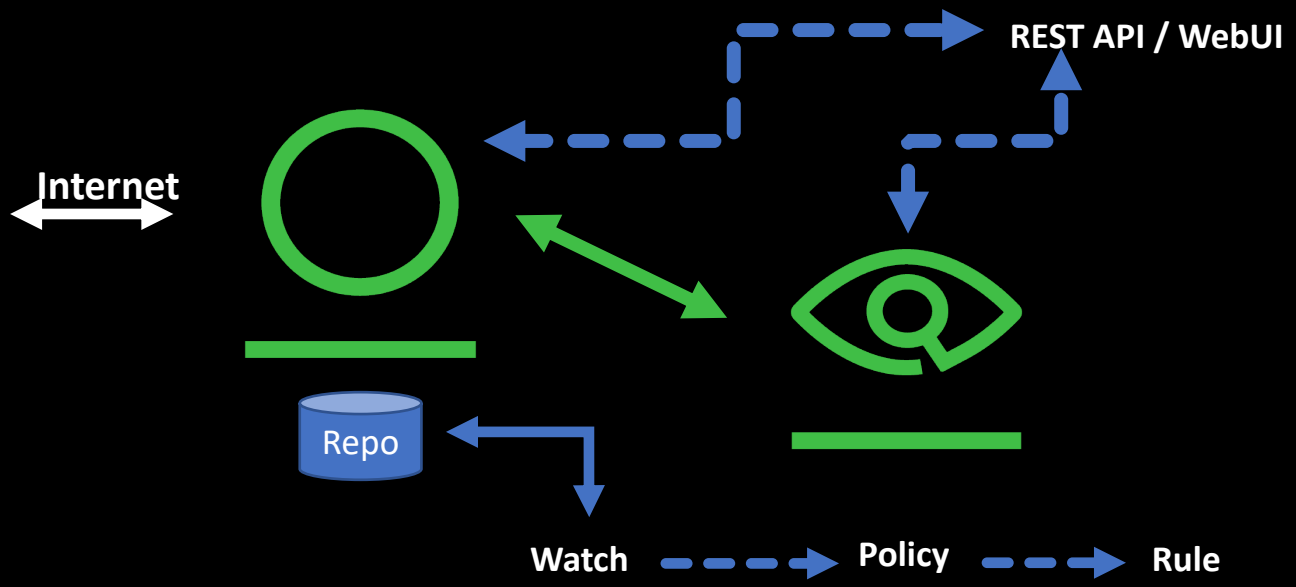


Security from scratch





Security from scratch



Rule: stateless definition

Policy: composition of Rules

Watch: combination between Resource to scan and Policies

Security from scratch

Application

Assume that we want to start with a POC

We start with an idea, some Use Cases
we want to write down as code

Don´t reinvent the wheel… use dependencies

Adding a new entry into the pom.xml

JFrog – XRay – Plugin will show known
- security- issues
- compliance- issues

Security from scratch

Linux / Docker

We have an application

Editing the Dockerfile

JFrog – XRay – will show known

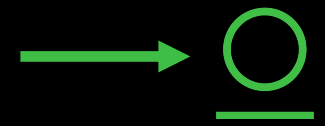
- security- issues
- compliance- issues

Security from scratch

Kubernetes - Universe

- Plain text passwords
- un-secure libs
- Bugs
- OS-Bugs / OS-Security Issues
- Outdated Docker Images
- Un-trusted Images

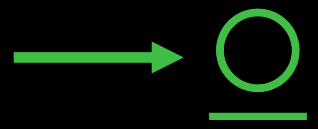
- Using in-build Kubernetes-Platform-Security-Features
- Reducing Kubernetes attack surface
  - Classical Proxy & Firewall
  - SSL Usage everywhere
  - Central Docker-Registry



  - Harden Docker-Images
  - Continuously re-scan
  - Update libs

Security from scratch

Kubernetes - Universe

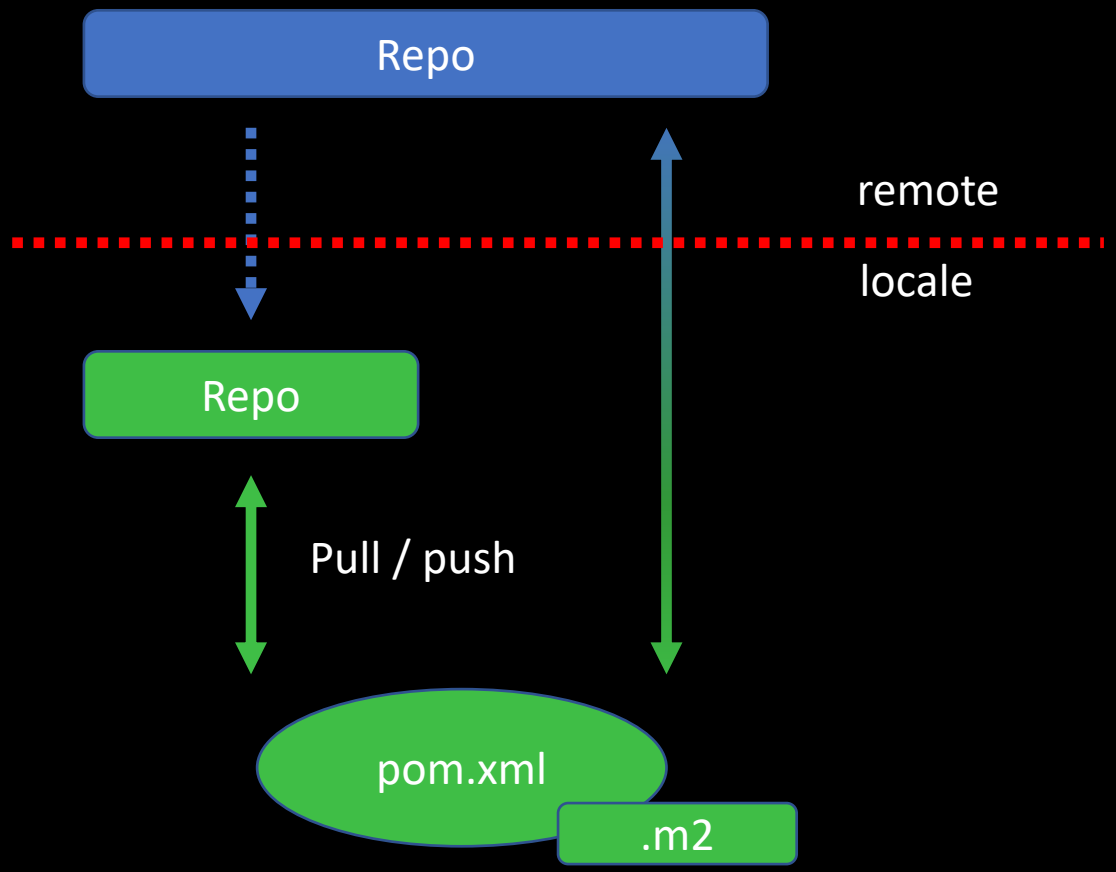
Central Docker-Registry 

What does Kubenab do?
Kubenab is a
Kubernetes Dynamic Admission Webhook
to enforce pulling of docker images
from private registry.

GitHub: **jfrog/kubenab**

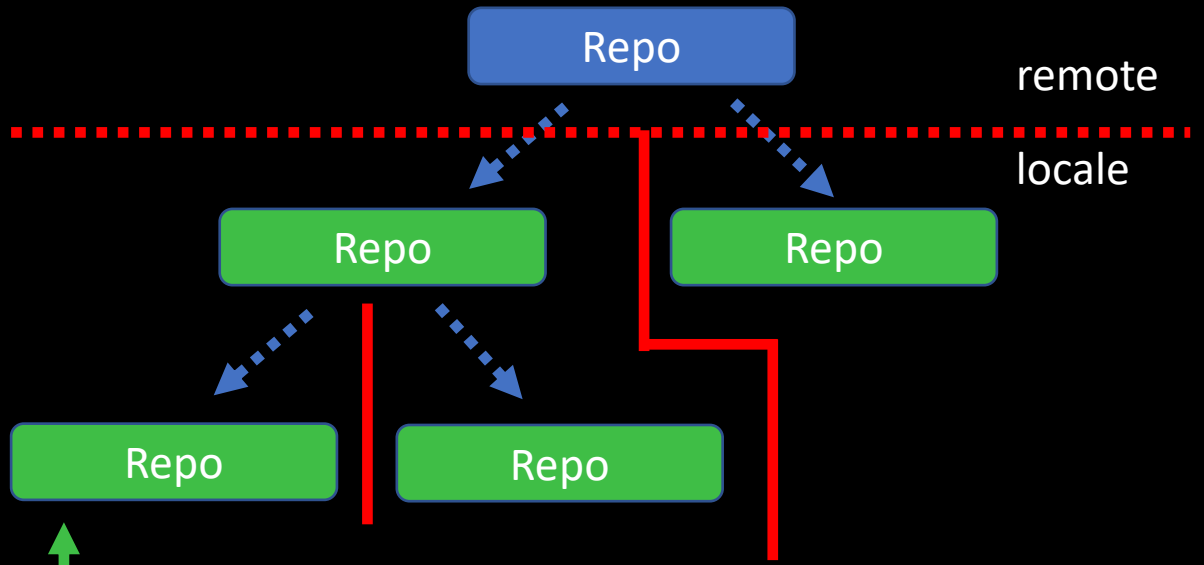
SpeedUp your CI and true Immutability

SpeedUp your CI and true Immutability



Caching of remote Artefacts

SpeedUp your CI and true Immutability



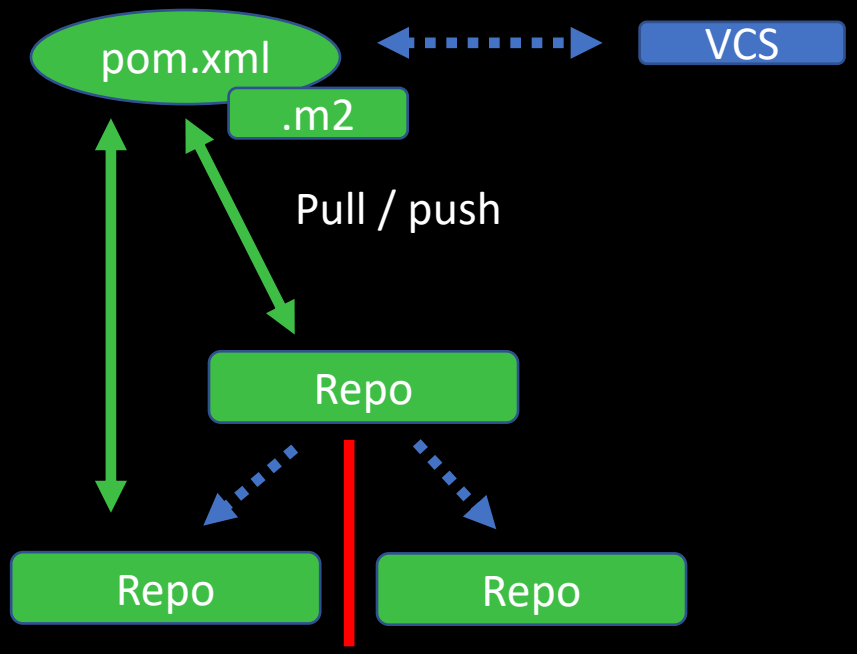
Tree – for Caching

A Child is a Superset

Content maintenance is difficult

How to share between sub-trees?

SpeedUp your CI and true Immutability



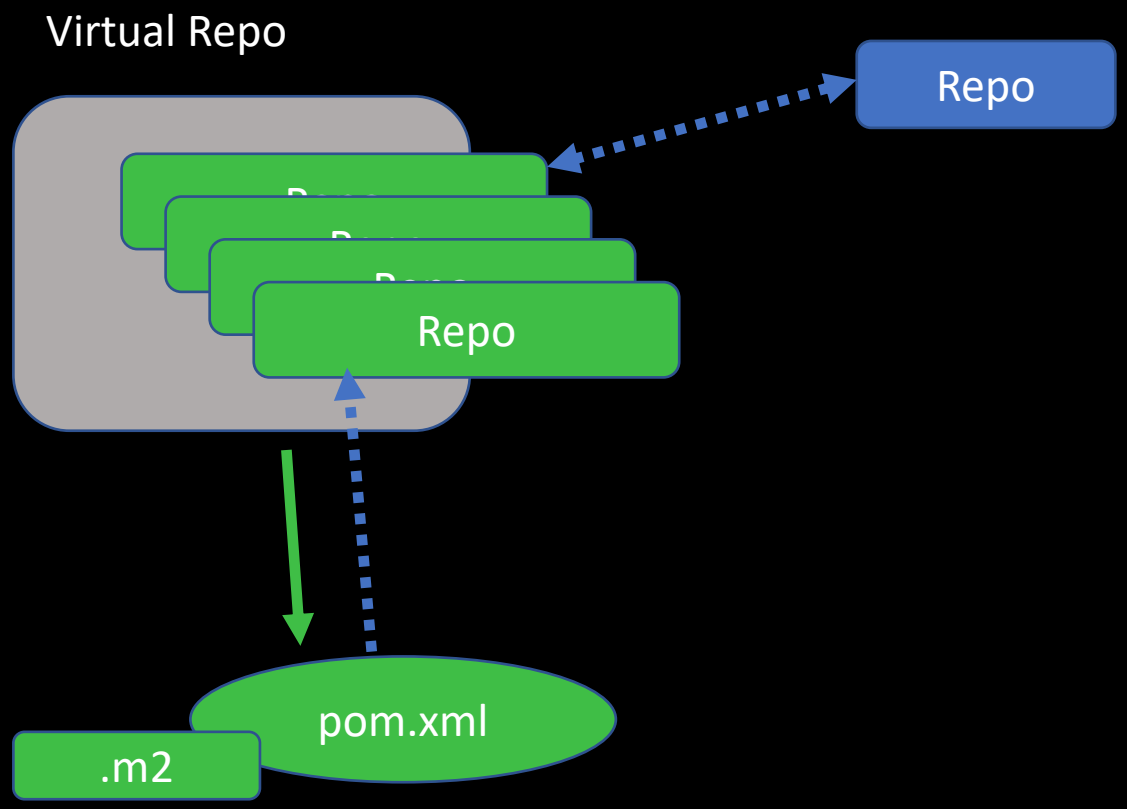
Tree – for Caching

A Child is a Superset

Content maintenance is difficult

How to share between sub-trees?

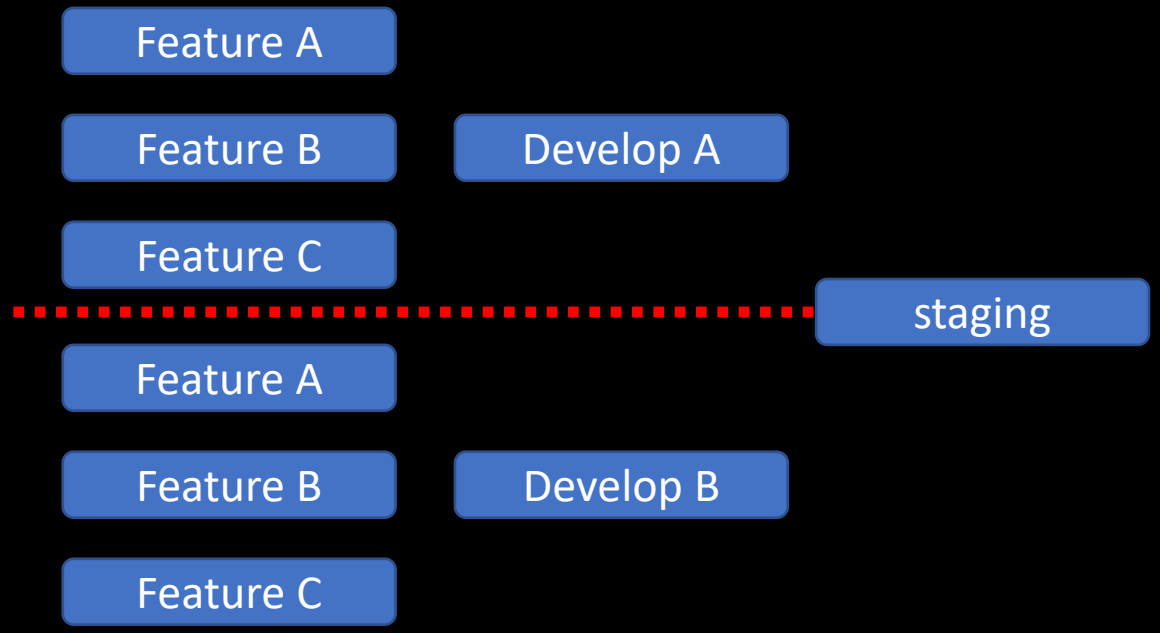
SpeedUp your CI and true Immutability



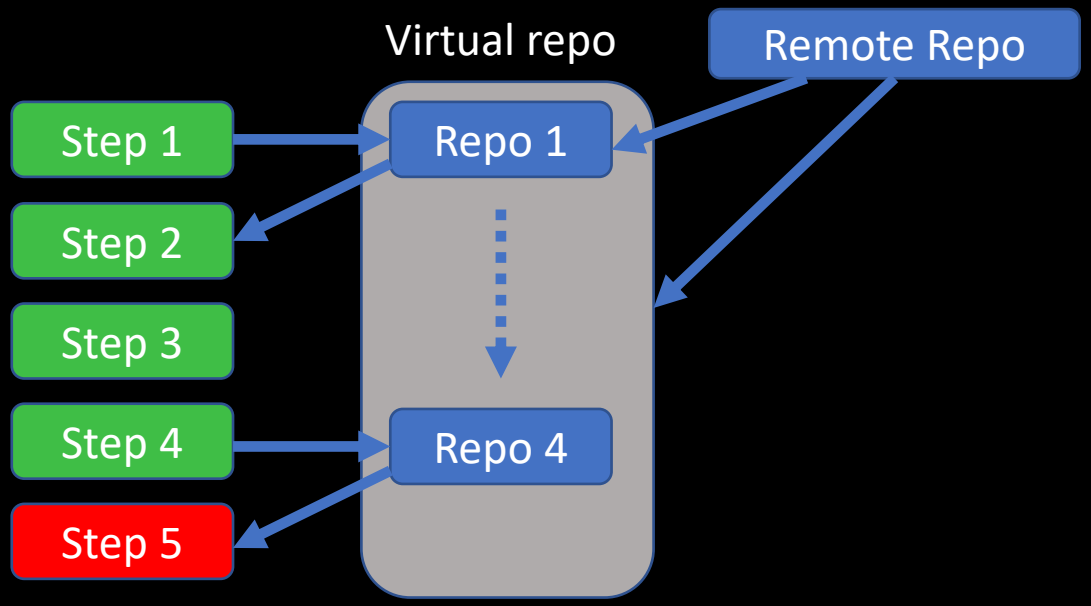
Transparent pull

Push with default target

SpeedUp your CI and true Immutability



SpeedUp your CI and true Immutability



Kill the Re-Build

Every Step will have one Repo

Virtual repo for Aggregation

Failed step X leads to deleted Repo X

The power of integration

The power of integration

Break a build

Notify with an email

Webhook → Activate workflows

REST API → Consume data in 3rd party products

Reporting

Compliance tooling

Dynamic workflows

Just try it

https://jfrog.com/start-free

- Go to the URL and start a trial
- Create a Remote Maven Repository
- Create a Watch for this repo
- Set maven to use the created Repository
- mvn clean ; mvn verify
- Check the Watch you created or
- Check the IDE plugin (JFrog – Plugin)

Who is speaking?

- **Sven Ruppert**
- DevSecOps, Java & Kotlin
- JFrog Inc
- **Twitter**: @SvenRuppert
- **eMail:** svenr@jfrog.com

Youtube: [DE] - bit.ly/Youtube-Sven
Youtube: [EN] - bit.ly/Outdoor-Nerd

THANK YOU!